METHODS, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR PARALLEL CORRELATION AND APPLICATIONS THEREOF

Inventors: Gregory S. Rawlins
Ray Kassel

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This patent application is a continuation-in-part of U.S. non-provisional application Ser. No. 09/987,193, titled, "Method and Apparatus for a Parallel Correlator and Applications Thereof," filed November 13, 2001, which claims priority to U.S. provisional application Ser. No. 60/248,001, titled, "Method and Apparatus for a Parallel Correlator and Applications Thereof," filed November 14, 2000, both of which are incorporated herein by reference in their entireties.

[0002] The following application of common assignee is related to the present application, and is herein incorporated by reference in its entirety: U.S. non-provisional application, "Method and System for Down-Converting an Electromagnetic Signal, Transforms for Same, and Aperture Relationships," Ser. No. 09/550,644, filed April 14, 2000.

BACKGROUND OF THE INVENTION

Field of the Invention

[0003] The present invention relates to matched filters and correlators and, more particularly, to a novel fast correlator transform ("FCT") and to methods and systems for implementing same.

Related Art

[0004] Matched Filter Theory was introduced by D.O. North in 1943. Others such as Van Vleck, Middleton, Weiner, and Turin, have added to the body of knowledge and application, ranging from RADAR to SONAR and communications. Although the basic theory was originally applied to continuous time systems and linear networks, the theory has been augmented to absorb discrete sampled filters and correlator operations. Indeed Paul Green pointed out in the June 1960 IRE Transactions on Information Theory, Matched Filter Issue:

> "Correlation detection was studied at first as a separate subject, but the equivalence of the two operations (meaning matched filters and correlators) was soon appreciated."

[0005] IRE Transactions on Information Theory, New York, NY: Professional Group on Information, Institute of Radio Engineers, June, 1960, incorporated herein by reference in its entirety.

[0006] More recently Van Whalen and Blahut as well as others have provided proofs of mathematical equivalence of correlation or matched filtering based on a change of variables formulation.

[0007] With the surge of DSP and proliferation of VLSI CMOS circuits as well as the universal push for software radios, many discrete matched filter realizations have evolved. The most notable digital implementation is the Transversal or Finite Impulse Response Filter which may realize the time flipped impulse response of the waveform to be processed or utilized as a correlator, both which produce the equivalent result at the optimum observation instant.

[0008] A particular concern arises when multiple filtering operations are required, concurrently, as is the case for parallel correlators. The complexity, clock speeds and signal flow control typically increase cost, size, and power.

[0009] What are needed are improved methods and systems for performing matched filtering and/or correlating functions, including concurrent and/or parallel correlations.

SUMMARY OF THE INVENTION

[0010] The present invention is directed to methods and systems for performing matched filtering and/or correlating functions, including concurrent and/or parallel correlations. More particularly, the present invention is directed to a fast correlator transform (FCT) algorithm and methods and systems for implementing same. The invention is useful for, among other things, correlating an encoded data word (X_0-X_{M-1}) with encoding coefficients (C_0-C_{M-1}), wherein each of (X_0-X_{M-1}) is represented by one or more bits and each coefficient is represented by one or more bits, wherein each coefficient has k possible states, and wherein M is greater than 1.

[0011] In accordance with the invention, X_0 is multiplied by each state ($C_{0(0)}$ through $C_{0(k-1)}$) of the coefficient C_0, thereby generating results $X_0C_{0(0)}$ through $X_0C_{0(k-1)}$. This is repeated for data bits (X_1-X_{M-1}) and corresponding coefficients (C_1-C_{M-1}), respectively. The results are grouped into N groups. Combinations within each of said N groups are added to one another, thereby generating a first layer of correlation results.

[0012] The first layer of results is grouped and the members of each group are summed with one another to generate a second layer of results. This process is repeated as necessary until a final layer of results is generated. The final layer of results includes a separate correlation output for each possible state of the complete set of coefficients (C_0-C_{M-1}). The results in the final layer are compared with one another to identify the most likely encoded data word.

[0013] In an embodiment, the summations are pruned to exclude summations that would result in invalid combinations of the encoding coefficients (C_0-C_{M-1}). In an embodiment, substantially the same hardware is utilized for processing in-phase and quadrature phase components of the data word (X_0-X_{M-1}). In an embodiment, the coefficients (C_0-C_{M-1}) represent real numbers. In an alternative embodiment, the coefficients (C_0-C_{M-1}) represent complex numbers. In an embodiment, the coefficients (C_0-C_{M-1}) are represented with a single bit. Alternatively, the coefficients (C_0-C_{M-1}) are represented with multiple bits (e.g., magnitude). In an embodiment, the coefficients (C_0-C_{M-1}) represent a cyclic code keying ("CCK") code set substantially in accordance with IEEE 802.11 WLAN standard.

[0014] Further features and advantages of the invention, as well as the structure and operation of various embodiments of the invention, are described in detail below with reference to the accompanying drawings. It is noted that the invention is not limited to the specific embodiments described herein. Such embodiments are presented herein for illustrative purposes only. Additional embodiments will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

BRIEF DESCRIPTION OF THE DRAWINGS/FIGURES

[0015] The present invention will be described with reference to the accompanying drawings. The drawing in which an element first appears is typically indicated by the leftmost digit(s) in the corresponding reference number.

[0016] FIG. 1 is a block diagram of an example discrete transversal matched filter or correlator, in which the present invention can be implemented.

[0017] FIG. 2A is an expanded view of a summation function 102 illustrated in FIG. 1.

[0018] FIG. 2B is a block diagram of a FCT kernel implemented in an 802.11 demodulator 210 in accordance with an aspect of the present invention.

[0019] FIG. 3A illustrates example correlation kernels, in accordance with an aspect of the present invention.

[0020] FIG. 3B illustrates example first and second layers of an FCT processing hierarchy in accordance with an aspect of the invention.

[0021] FIG. 3C illustrates an example signal flow diagram for the first and second layers illustrated in FIG. 3B.

[0022] FIG. 4A illustrates a conventional parallel correlator approach.

[0023] FIG. 4B illustrates an example matrix form of coefficients for a parallel correlator in accordance with an aspect of the present invention.

[0024] FIG. 5 illustrates an example complex fast Hadamard Transform.

[0025] FIG. 6 illustrates an example parallel magnitude compare operation in accordance with an aspect of the invention.

[0026] FIG. 7 illustrates an example final layer of an FCT processing hierarchy in accordance with an aspect of the invention.

[0027] FIG. 8 illustrates an example process flowchart for implementing an FCT algorithm in accordance with an aspect of the present invention.

[0028] FIG. 9 is a signal path diagram for an example CCK decoder output trellis, including an I signal path 902 and a Q signal path 904.

[0029] FIG. 10 is a block diagram of an inverted channel detection circuit 1000 that can be used to determine if one of the channels was inverted.

[0030] FIG. 11 is a block diagram of a complex correlation in accordance with equation 12.

[0031] FIG. 12 is a block diagram of an example FCT 1200 in accordance with the present invention.

[0032] FIG. 13 is a block diagram of another example FCT 1300 in accordance with the present invention, including stages 1a, 1b, 2a, 2b, and 3.

[0033] FIG. 14 is an example block diagram of stage 1a of the FCT 1300.

[0034] FIG. 15 is an example block diagram of stage 1b of the FCT 1300.

[0035] FIG. 16 is an example block diagram of stage 2a of the FCT 1300.

[0036] FIG. 17 is an example block diagram of stage 2b of the FCT 1300.

[0037] FIG. 18 is a block diagram of an example system 1800 that calculates A_n values.

[0038] FIG. 19 is a block diagram of connection stages for an example simplified architecture FCT 1900, including stages 1a, 1b, 2a, 2c, and 3.

[0039] FIG. 20 is an example block diagram of stage 1a of the FCT 1900.

[0040] FIG. 21 is an example block diagram of stage 1b of the FCT 1900.

[0041] FIG. 22 is an example block diagram of stage 2a of the FCT 1900.

[0042] FIG. 23 is an example block diagram of stage 2b of the FCT 1900.

[0043] FIGS. 24A and B are an example block diagram of stage 3 of the FCT 1900.

[0044] FIG. 25 illustrates an example structure 2500 for implementing equation 17.

[0045] FIG. 26 is another example block diagram of stages of the FCT 1900.

[0046] FIG. 27 is another example block diagram of stages of the FCT 1900.

DETAILED DESCRIPTION OF THE INVENTION

TABLE OF CONTENTS

I. Introduction

[0047] FIG. 1 is a block diagram of an example discrete transversal matched filter or correlator 100, also referred to herein as a finite impulse response ("FIR") filter 100. The FIR filter 100 includes a set of multipliers 104 that multiply data words X_i by coefficients C. The FIR filter 100 also includes a final accumulate or summation function 102. The matched filter 100 implements the following discrete sampling equation;

$$y_i = \sum_{k=0}^{n-1} c_k x_{(i-k)} \qquad \text{(Eq. 1)}$$

[0048] In Eq. 1, X_i are typically derived from a sampling device such as an A/D converter and can be represented as signed magnitude, two's complement, or other binary representation in parallel bit format. Likewise the multiplier tap coefficients $C_0 \ldots C_n$ can be 1 bit values or k bit (soft) values depending on the application. In the case of soft value implementations, the multiplier functions can represent considerable hardware or algorithmic complexity. The final summation can be implemented using discrete adder technologies or floating point processors. The entire architecture can be implemented in hardware, software, or a hybrid of each.

[0049] A particular concern arises when multiple filtering operations are required, concurrently, as is the case for parallel correlators. The complexity, clock speeds and signal flow control typically increase cost, size, and power. Hence, efficient architectures are often pursued to streamline the implementation and thereby differentiate product offerings.

[0050] The present invention is directed to a novel fast correlator transform ("FCT") algorithm that reduces the number of additions for parallel correlation, as compared to conventional techniques. The present invention is also directed to methods and systems for implementing the novel FCT algorithm. Conventional techniques are typically of less universal application and restricted to certain signaling schemes. The present invention, on the other hand, is applicable to a variety of parallel matched filter and/or correlator operations.

[0051] The present invention is as efficient or more efficient than the Fast Walsh Transform, currently applied as the "state of the art," and is more universally applicable to signaling schemes employing higher order signaling spaces such as

MQAM, CDMA, etc. In addition, classical multi-path processor algorithms are more easily applied using the classical correlator/matched filter kernel, in accordance with the present invention, rather than the conventional modified Fast Walsh Transform.

[0052] The present invention can be implemented in hardware, software, firmware, and/or combinations thereof. For example, and without limitation, the invention, or portions thereof, can be implemented in hardware logic. The invention, or portions thereof, can also be implemented in software imbedded in a computer readable medium (i.e., a computer program product), which, when executed on a computer system, causes the computer system to perform in accordance with the invention.

[0053]

II. Example Environment: 802.11

[0054] The present invention is described herein as implemented in an example environment of an IEEE 802.11b 11 MBPS physical layer signaling scheme. IEEE 802.11 is a well-known communications standard and is described in, for example, "Medium Access Control (MAC) and Physical (PHY) Specifications," ANS/IEE Std 802.11, published by IEEE, (1999Ed.), and incorporated herein by reference in its entirety.

[0055] The present invention is not, however, limited to the IEEE 802.11 communications standard. Based on the description herein, one skilled in the relevant art(s) will understand that the present invention can be implemented for a variety of other applications as well. Such other applications are within the spirit and scope of the present invention.

[0056] A distinction is made herein between a basic correlator kernel and a full demodulator. A general form of a demodulator for IEEE 802.11 typically requires four correlator banks operating essentially in parallel. A fast correlator transform ("FCT") kernal, in accordance with the invention, typically includes similar structure, plus complex additions and subtractions from two in-phase and two quadrature-phase banks to accomplish the demodulation for IEEE 802.11. The Walsh transform, as a comparison, accounts for these additional adds and subtracts by having them built into its algorithm.

[0057] The 802.11 signaling scheme of interest is based on Cyclic Code Keying ("CCK") derived from Walsh/Hadamard functions. A restricted set within the available coding space was selected so that the Fast Walsh Transform could be utilized to implement an efficient correlator architecture. Originally, both Harris and Lucent could not figure out how to apply a classical parallel matched filter or correlator, efficiently, to process the waveforms. The current coding space provides for 64 code words. Harris erroneously predicted that a classical parallel matched filter approach would require 8 x 64 - 512 complex additions since each code word is 8 bits, on I and Q and there are 64 code words. However, the true estimate is 7 x 64 = 448 complex additions as illustrated in the example 8-way adder tree illustrated in FIG. 2A.

[0058] FIG. 2A is an expanded view of the final accumulate or summation function 102 in FIG. 1, part of the FIR filter 100. Notice that only 7 adders are required for the example implementation. Nevertheless, 448 complex additions represent a significant number of operations. Lucent, Harris/Intersil, and Alantro apply the Fast Walsh Transform ("FWT") to the CCK code set to reduce the correlation operation down to 112 complex multiplies due to the restriction placed on the code set.

[0059] The FWT is actually more of a decoder than a correlator. It reduces to the performance of a correlator for this specific application if the coefficients in the butterfly branches are weighted with simple hard coded values, i.e., 1, -1, j, -j. The imaginary numbers are included for the case of complex signaling.

[0060] The FCT algorithm, according to the present invention, is truly a correlation or matched filter operation and can be applied with soft value weighting or hard value weighting. Furthermore, the present invention is intuitively satisfying, possessing direct correspondence of matched filter tap weights or coefficients maintained throughout the hierarchical structure. This permits easy extension of the matched filter concept to accommodate channel equalization, MLSE, and other adaptive applications.

[0061] FIG. 2B is a block diagram of an example FCT kernel implemented in an 802.11 demodulator 210. The 802.11 demodulator 210 includes a bank of FCT kernels 212. The demodulator 210 can be reduced to 110 complex operations, with 78 of the complex operations allocated to the bank of FCT kernels 212. Additional simplification can be obtained by exploiting redundancy in the algorithm, as described below.

III. Fast Correlator Transform and Correlator Kernels

[0062] The available coding space for a 16 bit word is $2^{16} = 65536$. CCK whittles this space down to a code set formed by 8 in-phase ("I") chip and 8 quadrature phase ("Q") chip complex symbols. 64 code words are retained, carrying 6 bits of information. 2 more bits are contained in the complex QPSK representation, for a total of 8 bits of information.

[0063] Suppose then that 8 samples of an input vector $X_0, X_1, \ldots X_7$ are associated with optimal sampling instants from the output of a chip matched filter. Each of these samples would normally be weighted by the coefficients $C_0 \ldots C_7$ then assimilated as illustrated in FIGS. 1 and 2A.

[0064] In the 802.11 example, $C_0 \ldots C_7$ correspond to samples of the CCK code set symbols. The unknowns $X_0 \ldots X_7$ are noisy input samples from which a specific code must be extracted. Conceptually, 64 such complex filtering operations should be performed concurrently since a specific code word cannot be predicted a priori. The largest output from the parallel bank of correlators would then be selected as the most likely code word correlation match at that particularly symbol sample time.

[0065] In accordance with the present invention, a general philosophy for correlation is imposed for partitioned segments of code words. In an embodiment, the code word correlations are divided into sub-sets. In the illustrated example embodiment, the code word correlations are divided into sample pairs. The present invention is not, however, limited to this example embodiment. In other embodiments, the code word correlations are divided into triplets, quintuplets, and/or any other suitable sub-sets.

[0066] Combinations of the code word correlation sub-sets are then provided to correlation kernels. FIG. 3A illustrates example correlation kernels 302a-302d, in accordance with an aspect of the present invention.

[0067] The correlation kernels 302a-302d represent all or substantially all possible correlation options for the first 2 samples X_0, X_1. In a similar fashion the remaining groupings of double samples (X_2, X_3), (X_4, X_5), (X_6, X_7) will also spawn their own set of 4 correlation kernels.

[0068] The number of separate correlation kernels spawned is determined by the number of correlated samples per summation, the number of correlation variations possible, and, in an embodiment, the number of invalid combinations of correlated samples. In the present example, each coefficient has two possible states (i.e., hard value weighting). Thus each subset correlation generates 4 outputs. In alternative embodiments, soft weighting is implemented, where each coefficient is represented with multiple bits (e.g., magnitude representation).

[0069] In an embodiment, the number of correlation operations associated with binary antipodal signaling in accordance with the present invention is implemented in accordance with Eq. 2.

$$N_k = \frac{n!}{r!(n-r)!} - L \qquad \text{(Eq. 2)}$$

[0070] The result for the example environment described above, using 2 input summers, is shown in Eq. 3:

$$N_k = \frac{4!}{2!(2)!} - 2 \qquad \text{(Eq. 3)}$$

wherein:

n is the number of uniquely available summer inputs;

r is the number of summing inputs per kernel; and

L is the number of invalid combinations.

[0071] N_k is thus the number of correlation kernels and therefore the number of adders or summers required. Groupings of two correlation samples provide for convenient binary expansion. As stated above, however, the present invention is not limited to groupings of two.

[0072] The term L represents the number of invalid or disallowed combinations. For instance, X_0C_0 and $-X_0C_0$ is an invalid combination when added and therefore can be subtracted from the total number of combinations. In an embodiment, 3 way adders or summers are utilized. In other embodiments, other algorithmic partitioning is utilized. For the current example, partitioning in powers of 2 is convenient and attractive in terms of potential hardware implementation.

[0073] FIG. 3B illustrates example first and second layers 304 and 306, respectively, of an FCT processing hierarchy in accordance with an aspect of the invention.

[0074] The second layer 306 includes 32 additions 308, of which 308a through 308p are illustrated, to process combinations of correlations from the first layer 304 of correlation kernels. The first layer 304 includes 16 additions related to the first 4 sample correlations $X_0C_0 \ldots X_3C_3$, and 16 additions related to the 2nd 4 sample correlations. Hence, the first layer 304 of kernels includes 16 adders 302 and the second layer possesses 32 adders 308. Once the second layer 306 is accomplished, each term that results includes 4 correlation components.

[0075] Note that 4 unique samples $X_0 \ldots X_3$ at the system input spawns 2^4 unique 4-tuple correlations at the second layer 306 of kernel processing. The corresponding number of additions is calculated for the 4 sample correlation sequences from Eq. 4:

$$N = \frac{8!}{2!6!} - 2\left(\frac{4!}{2!2!}\right) = 16 \qquad \text{(Eq. 4)}$$

[0076] At this point it is important to realize that all that is required is one more level of processing to obtain correlation terms consisting of 8 components which represents a full length correlation. However, it must also be recognized that there are 16 upper 4-tuple correlations as well as 16 lower 4-tuple correlations, which if exploited for all combinations in this case would yield 256 addition operations! Fortunately the CCK code set is well defined and possesses only 64 valid 8 chip component correlations. Hence, the final layer, illustrated in FIG. 7, is pruned to perform only 64 unique addition operations. Thus, a total (upper bound) number of adders used for the algorithm is:

16 (first hierarchical layer) + 32 (second layer) + 64 (third layer) = 112

[0077] This is a remarkable result because a conventional parallel matched filter or correlator bank would require 448 complex additions. Theoretically, 112 is the upper bound. However, in practice, the Trellis may be pruned to a maximum of 78 additions on the I and 78 and the Q.

[0078] FIG. 3C illustrates an example signal flow diagram for the FCT algorithm through the first 2 layers 304 and 306. In accordance with the example above, there are 8 input samples and 32 output correlation options through the first 2 layers 304 and 306. Correlation combinations from the upper and lower 16 4-tuples provide a final trellis with 64 8-tuple options, each representing a different CCK symbol. The option having the highest output is selected during each symbol cycle as the most likely CCK symbol. In the example embodiment, the correlation operation utilizes I and Q matched filters since the symbols are complex.

IV. Mathematical Formulation

[0079] In an embodiment, a receiver in accordance with the present invention receives a waveform and an analog-to-digital converter function samples the received waveform and converts the waveform down to baseband. The received sampled waveform may be represented from Eq. 5:

$$X_i = S_i + N_i \qquad \text{(Eq. 5)}$$

[0080] Where S_i represents samples from the received signal and N_i represent noise sampled from an average white Gausian noise ("AWGN") process. This equation does not account for multi-path. The samples can be considered as complex, with I and Q components. The receiver output can be represented by Eq. 6:

$$Y_i = \sum_{k=0}^{n-1} C_k X(i-k) \qquad \text{(Eq. 6)}$$

[0081] The coefficients, C_k, are considered constant for the nominal case of an AWGN channel. "n" is the FIR filter or correlator depth. For a case of m correlators operating on X_i in parallel, Eq. 6 becomes Eq. 7:

$$Y_{i,m-1} = \sum_{k=0}^{n-1} C_{k,m-1} X(i-k) \qquad \text{(Eq. 7)}$$

[0082] The mth correlator branch then contains correlator coefficients uniquely related to that branch.

[0083] FIG. 4A illustrates a conventional parallel correlator approach and relates it to Eq. 7. The present invention breaks the sum over n-1 into smaller sums, typically, although not necessarily, sums of 2. The present invention applies all, or substantially all potential cross correlations and carries the 4 results forward to a subsequent level of processing. An example mathematical formulation of this operation is provided in Eq. 8;

$$Y(i)_{\ell,v,p,u} = \sum_{k=0}^{1} C_{k,\ell} X(i-k) + \sum_{k=2}^{3} C_{k,v} X(i-k) + \sum_{k=4}^{5} C_{k,p} X(i-k) + \sum_{k=6}^{7} C_{k,u} X(i-k) \cdots \qquad \text{(Eq. 8)}$$

[0084] Where $\ell, v, p,$ and u may be considered as indices to select different coefficients. All indices should be considered in terms of a final valid code word correlation. In the 802.11 case, 256 correlation sequences are defined by Eq. 8, but the options are sifted to correspond only to valid CCK code words. FIG. 4B illustrates an example matrix form of coefficients for a parallel correlator according to the present invention.

[0085] The coefficients all take on the values of +/-1 for the examples herein. The indices are permitted to contain zeros for consistency with the original FIR formulation. The FCT sub-matrices however are simply;

$$C_{k,\ell} = C_{k,v} = C_{k,p} = C_{k,u} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \text{(Eq. 9)}$$

[0086] The indices $\ell,\nu,\mathbf{p},\mathbf{u}$ are manipulated specifically to match the coefficients to the desired code words at the $\mathbf{Y(i)}_{\ell,\nu,\mathbf{p},\mathbf{u}}$ outputs. The indices also uniquely specify the trajectory through the signal flow path. Breaking up the original parallel matched filter coefficient matrix into the smaller 2x2 matrix permits the FCT algorithm to exploit redundant correlation operations.

V. Comparisons to the Hadamard Transform

[0087] An FCT algorithm trellis, in accordance with the present invention, is described above. A corresponding length 4 complex fast Hadamard Transform is illustrated in FIG. 5. As with the FCT algorithm, there are two such trellis structures corresponding to 8 input samples and 32 outputs. The 32 outputs include two 16 wide 4-tuple groupings, which are then utilized in combinations to produce the final 54 8-tuple correlations.

[0088] There are distinct differences between the two algorithms. For example, the FCT algorithm can function on arbitrary correlation words, even in matched filter scenarios, while the FWT requires certain signal coding structure. Also, the FCT algorithm permits greater efficiency because the number of adds may be tailored for a specific code set or application.

[0089] Harris and Lucent advertise an efficiency of 112 complex additions for CCK demodulation, which amounts to 2x112 additions. The bounding case for 64 arbitrary correlations with the FCT was shown to be 112, maximum. It turns out that the specific case of CCK may be accommodated using a pruned FCT algorithm having 78 additions on the in-phase correlation band and 78 additions on the quadrature correlation bank. An additional 36 add/subtract operations are required to complete the demodulator, for a total of 114 complex operations, versus 112 for the FWT. Additional reductions can be obtained to reduce the demodulator to approximately 100 complex add/subtract operations, as described below.

VI. Maximum Likelihood Decoding (AWGN, no Multipath)

[0090] The modified Fast Walsh/Hadamard Transform implements a complex trellis decoder for which maximum scores may be compared at the trellis output. Thus there are 64 distinct outputs of the trellis which are preferred based on Euclidean distance calculations along the trellis trajectories. Only certain specific trajectories are considered up through the second tier of the trellis. The distance properties for the decoding trellis are also Euclidean for the in-phase and quadrature phase correlators. However, it is important to realize that the total distance should be calculated within the complex plane rather than simply on I independently of Q. That is, scoring is based on Eq. 10.

$$\text{Distance} \equiv \sqrt{I_{score}^2 + Q_{score}^2} \qquad \text{(Eq. 10)}$$

[0091] This comes from the fact that there are pairs of I and Q chip code words which are dependent. That is the nature of the complex Walsh-Hadamard codes. Fortunately, a sufficient statistic exists which does not require the square root operation. Simply calculating the sum of the squares or estimating the vector magnitude will suffice. In this manner then the total distance or weighting through complex space is calculated. The largest output out of the 64 complex operations (weighting scenario) then becomes the most likely 8 chip complex code.

A. Magnitude Comparator

[0092] In order to determine which code symbol was most likely encoded, a magnitude compare operation is performed on the outputs from the summer 102 (FIG. 1). A variety of types of magnitude compare operations can be performed.

[0093] FIG. 6 illustrates an example parallel magnitude compare operation in accordance with an aspect of the invention. In operation, the I and Q inputs, 8 bits wide each, for example, are squared and summed at the correlator outputs to form 64 scores. These 64 scores are compared and the largest result selected as the maximum likelihood symbol estimate. Each of the 64 outputs are assigned their corresponding chip code-to-6-bit data map. An additional di-bit is decoded from the differential phase decoder. In an embodiment, the squaring operation results in 16 bit output value when the inputs from each I and Q correlator are truncated to an extent reasonable to minimize the compare tree. In an embodiment, a parallel compare tree utilizes $\log_2(64)-1$ compares to obtain the most likely result.

[0094] In an embodiment, the magnitude compare operation illustrated in FIG. 6 utilizes a flag at each level of compare to indicate the winning local score at that level. The winning local score can be traced from the output back to one of the 64 original input correlation scores to decide which 6-bit word is most likely. In an embodiment, outcomes of tie scores at one or more levels are arbitrarily determined. In an embodiment, magnitude compare operations are performed with an adder/subtractor to create the result C=A-B, where A and B are inputs.

[0095] Another magnitude compare technique that can be utilized is referred to herein as space slicing, which includes the steps of examining the MSB of the correlator outputs, and throwing away results not active in the MSB. If none are active in the MSB then the next MSB is compared, so on and so forth. Any surviving correlator outputs are compared in the next most significant MSB in succession until all necessary compares are exhausted. This technique is useful because it requires only 1-bit compares at each level down to the final compare. In an embodiment, 1 bit compares are performed with an exclusive OR gate. Generally, there is no deterministic way to predict the number of surviving compares which may be passed on to the next level, but the maximum number typically reduces by a factor of 2 at each level. This approach relies on a statistical distribution of scores, which may permit rapid sifting. If all of the distances are similar in magnitude then sifting typically requires more operations. For instance, if all 64 distance calculations/scores possess an active MSB then the first round of sifting will not eliminate any scores and all scores are then be compared in the next MSB. Although this is not likely to occur, it should be anticipated for associated hardware realization.

VII. Example Methods for Implementing the FCT Algorithm

[0096] FIG. 8 illustrates an example process flowchart 800 for implementing the FCT algorithm in accordance with an aspect of the present invention. For illustrative purposes, the flowchart 800 is described herein with reference to one or more of the drawing figures described above. The invention is not, however, limited to the examples illustrated in the drawings. Based on the description herein, one skilled in the relevant art(s) will understand that the invention can be implemented in a variety of ways.

[0097] The example process flowchart 800 illustrates a method for correlating an encoded data word (X_0-X_{M-1}) with encoding coefficients (C_0-C_{M-1}), wherein each of (X_0-X_{M-1}) is represented by one or more bits and each said coefficient is represented by one or more bits, wherein each coefficient has k possible states, wherein M is greater than 1.

[0098] The process begins with step 802, which includes multiplying X_0 with each state ($C_{0(0)}$ through $C_{0(k-1)}$) of the coefficient C_0, thereby generating results $X_0C_{0(0)}$ through $X_0C_{0(k-1)}$. This is illustrated, for example, in FIGS. 3A, 3B, and 3C just prior to the kernels 302A, B, C, and D.

[0099] Step 804 includes repeating step 802 for data bits (X_1-X_{M-1}) and corresponding coefficients (C_1-C_{M-1}), respectively. This is illustrated, for example, in FIGS. 3B, and 3C just prior to the kernels 302E through 302Q.

[00100] Step 806 includes grouping the results of steps 802 and 804 into N groups and summing combinations within each of said N groups, thereby generating a first layer of correlation results. This is illustrated, for example, in FIGS. 3A, 3B, and 3C by the kernels 302, and the resultant first layer of results 307.

[00101] Step 808 includes grouping the results of step 806 and summing combinations of results within each group to generate one or more additional layers of results, and repeating this process until a final layer of results includes a separate correlation output for each possible state of the complete set of coefficients (C_0-C_{M-1}). This is illustrated in FIG. 3C and FIG. 7, where the summers 306 generate a second layer 310, the FCT final output trellis 702 (FIG. 7) provides separate outputs for each possible state of the complete set of coefficients (C_0-C_{M-1}) in a final layer 704.

[00102] In an embodiment, steps 806 and 808 include the step of omitting summations that would result in invalid combinations of the encoding coefficients (C_0-C_{M-1}). This is illustrated in steps 806A and 808A. This also is illustrated, for example, in FIG. 7, wherein the second level of results 310 omits the following combinations:

$C_4C_5C_6(-C_7)$;
$C_4C_5(-C_6)(-C_7)$;
$(-C_4)C_5C_6(-C_7)$;
$(-C_4)C_5(-C_6)(-C_7)$;
$C_4(-C_5)C_6(-C_7)$;
$C_4(-C_5)(-C_6)(-C_7)$;
$(-C_4)(-C_5)C_6(-C_7)$; and
$(-C_4)(-C_5)(-C_6)(-C_7)$.

[00103] In this example, the omissions eliminate performing summations for combinations that are invalid in light of the CCK code or that result in null operation. In other embodiments, different combinations may or may not be omitted based on particular codes.

[0100] Step 810 includes comparing magnitudes of said separate correlation outputs, thereby identifying a most likely code encoded on said data word. This is illustrated, for example, in FIG. 6, by the example parallel magnitude compare operation.

[0101] In an embodiment, the process flowchart 800 further includes the step of performing steps (1) through (5) using substantially the same hardware for in-phase and quadrature phase components of the data word (X_0-X_{M-1}).

[0102] In an embodiment, the coefficients (C_0-C_{M-1}) represent real numbers. In an alternative embodiment, the coefficients (C_0-C_{M-1}) represent complex numbers.

[0103] In an embodiment, the coefficients (C_0-C_{M-1}) are represented with a single bit. Alternatively, the coefficients (C_0-C_{M-1}) are represented with multiple bits (e.g., magnitude).

[0104] In an embodiment, the coefficients (C_0-C_{M-1}) represent a cyclic code keying ("CCK") code set substantially in accordance with IEEE 802.11 WLAN standard, illustrated in the tables below.

[0105] In an embodiment, as illustrated in one or more of the prior drawing figures, M equals 8, N equal 4, and the coefficients (C_0-C_{M-1}) have two states, plus and minus.

VIII. CCK Chip Code Words

[0106] Tables are provided below that illustrate source-input data symbols, 8-bits long $(d_0 - d_7)$, and corresponding in-phase and quadrature phase 8 chip symbols used for CCK. The complex chip notation is provided for reference. In addition the algorithm flow diagram 4-tuple sums are provided since the last level of flow diagram becomes complex and difficult to follow. $B_0 \ldots B_{31}$ are the 4-tuple intermediate correlation results relating to the signal flow diagrams presented for the correlator algorithm. Two branch 4-tuples form the final output result for each correlator. $B_0 \ldots B_{15}$ provide options for the first branch component to form a final output correlator 8-tuple while $B_{16} \ldots B_{31}$ provide the second branch component or second 4-tuple. For instance, Table 1 illustrates an example build-up:

Table 1

4-tuple Designator	4-tuple Coefficient Sequence
B_6	-1,1,1,-1
B_{28}	1,1,-1,-1
4-tuple Combination $B_6 + B_{28} \Rightarrow$	Final 8-tuple Correlator Coefficient Sequence $-1,1,1,-1,1,1,-1,-1$

[0107] Logical zeros become weighted by an arithmetic value, -1. In this manner the optimum correlator trajectory for a particular chip sequence is projected through the correlator trellis. The example above corresponds to the in-phase correlator waiting for an originally transmitted data sequence $d_0 \cdots d_7$ **of 0,0,1,0,1,0,1,0**. For this example, that particular branch represents a correlation provided in Eq. 11;

$$y_{42} = x_0(-1) + x_1(1) + x_2(1) + x_3(-1) + x_4(1) + x_5(1) + x_6(-1) + x_7(-1) \quad \text{(Eq. 11)}$$

[0108] $x_0 \ldots x_7$ represent corrupted or noisy input signal samples. When the x_i match the coefficients significantly then that 8-tuple (1 of 64) output possesses maximum weighting and is declared most likely by the magnitude comparator. Another strategy seeks to minimize the distance between the x_i and c_i. The process is similar in either case.

[0109] Table 2 illustrates example in-phase and quadrature 4-tuple combinations. It is noted that the examples provided in Tables 1 and 2 are provided for illustrative purposes and are not limiting. Other specific examples will be apparent to persons skilled in the relevant arts based on the teachings herein, and such other examples are within the scope and spirit of the invention.

Table 2

d0 d1 d2 d3 d4 d5 d6 d7		In phase	4-tuple Combination	Quadrature	4-tuple Combination	Complex
D_0	00000000	11101101	$B_2 + B_{20}$	11101101	$B_2 + B_{20}$	111-111-11
D_1	00000001	00011101	$B_{13} + B_{20}$	11101101	$B_2 + B_{20}$	jjj-j11-11
D_2	00000010	00011101	$B_{13} + B_{20}$	00011101	$B_{13} + B_{20}$	-1-1-1111-11
D_3	00000011	11101101	$B_2 + B_{20}$	00011101	$B_{13} + B_{20}$	-j-j-jj11-11
D_4	00000100	00100001	$B_{14} + B_{23}$	11101101	$B_2 + B_{20}$	jj1-1jj-11
D_5	00000101	00010001	$B_{13} + B_{23}$	00101101	$B_{14} + B_{20}$	-1-1j-jjj-11

d0 d1 d2 d3 d4 d5 d6 d7		In phase	4-tuple Combination	Quadrature	4-tuple Combination	Complex
D_6	00000110	11010001	$B_1 + B_{23}$	00011101	$B_{13} + B_{20}$	-j-j-11jj-11
D_7	00000111	11100001	$B_2 + B_{23}$	11011101	$B_1 + B_{20}$	11-jjjj-11
D_8	00001000	00100001	$B_{14} + B_{23}$	00100001	$B_{14} + B_{23}$	-1-11-1-1-1-11
D_9	00001001	11010001	$B_1 + B_{23}$	00100001	$B_{14} + B_{23}$	-j-jj-j-1-1-11
D_{10}	00001010	11010001	$B_1 + B_{23}$	11010001	$B_1 + B_{23}$	11-11-1-1-11
D_{11}	00001011	00100001	$B_{14} + B_{23}$	11010001	$B_1 + B_{23}$	jj-jj-1-1-11
D_{12}	00001100	11101101	$B_2 + B_{20}$	00100001	$B_{14} + B_{23}$	-j-j1-1-j-j-11
D_{13}	00001101	11011101	$B_1 + B_{20}$	11100001	$B_2 + B_{23}$	11j-j-j-j-11
D_{14}	00001110	00011101	$B_{13} + B_{20}$	11010001	$B_1 + B_{23}$	jj-11-j-j-11
D_{15}	00001111	00101101	$B_{14} + B_{20}$	00010001	$B_{13} + B_{23}$	-1-1-jj-j-j-11
D_{16}	00010000	01000111	$B_7 + B_{17}$	11101101	$B_2 + B_{20}$	j1j-1j1-j1
D_{17}	00010001	00010111	$B_{13} + B_{17}$	01001101	$B_7 + B_{20}$	-1j-1-jj1-j1
D_{18}	00010010	10110111	$B_8 + B_{17}$	00011101	$B_{13} + B_{20}$	-j-1-j1j1-j1
D_{19}	00010011	11100111	$B_2 + B_{17}$	10111101	$B_8 + B_{20}$	1-j1jj1-j1
D_{20}	00010100	00000011	$B_{15} + B_{19}$	01100101	$B_6 + B_{21}$	-1jj-1-1j-j1
D_{21}	00010101	10010011	$B_9 + B_{19}$	00000101	$B_{15} + B_{21}$	-j-1-1-j-1j-j1
D_{22}	00010110	11110011	$B_0 + B_{19}$	10010101	$B_9 + B_{21}$	1-j-j1-1j-j1
D_{23}	00010111	01100011	$B_6 + B_{19}$	11110101	$B_0 + B_{21}$	j11j-1j-j1
D_{24}	00011000	10001011	$B_{11} + B_{18}$	00100001	$B_{14} + B_{23}$	-j-1j-1-j-1-j1
D_{25}	00011001	11011011	$B_1 + B_{18}$	10000001	$B_{11} + B_{23}$	1-j-1-j-j-1-j1
D_{26}	00011010	01111011	$B_4 + B_{18}$	11010001	$B_1 + B_{23}$	j1-j1-j-1-j1
D_{27}	00011011	00101011	$B_{14} + B_{18}$	01110001	$B_4 + B_{23}$	-1j1j-j-1-j1
D_{28}	00011100	11001111	$B_3 + B_{16}$	10101001	$B_{10} + B_{22}$	1-jj-11-j-j1
D_{29}	00011101	01011111	$B_5 + B_{16}$	11001001	$B_3 + B_{22}$	j1-1-j1-j-j1

d0 d1 d2 d3 d4 d5 d6 d7		In phase	4-tuple Combination	Quadrature	4-tuple Combination	Complex
D_{30}	00011110	00111111	$B_{12} + B_{16}$	01011001	$B_5 + B_{22}$	-1j-j11-j-j1
D_{31}	00011111	10101111	$B_{10} + B_{16}$	00111001	$B_{12} + B_{22}$	-j-11j1-j-j1
D_{32}	00100000	01000111	$B_7 + B_{17}$	01000111	$B_7 + B_{17}$	-11-1-1-1111
D_{33}	00100001	10110111	$B_8 + B_{17}$	01000111	$B_7 + B_{17}$	-jj-j-j-1111
D_{34}	00100010	10110111	$B_8 + B_{17}$	10110111	$B_8 + B_{17}$	1-111-1111
D_{35}	00100011	01000111	$B_7 + B_{17}$	10110111	$B_8 + B_{17}$	j-jjj-1111
D_{36}	00100100	10001011	$B_{11} + B_{18}$	01000111	$B_7 + B_{17}$	-jj-1-1-jj11
D_{37}	00100101	10111011	$B_8 + B_{18}$	10000111	$B_{11} + B_{17}$	1-1-j-j-jj11
D_{38}	00100110	01111011	$B_4 + B_{18}$	10110111	$B_8 + B_{17}$	j-j11-jj11
D_{39}	00100111	01001011	$B_7 + B_{18}$	01110111	$B_4 + B_{17}$	-11jj-jj11
D_{40}	00101000	10001011	$B_{11} + B_{18}$	10001011	$B_{11} + B_{18}$	1-1-1-11-111
D_{41}	00101001	01111011	$B_4 + B_{18}$	10001011	$B_{11} + B_{18}$	j-j-j-j1-111
D_{42}	00101010	01111011	$B_4 + B_{18}$	01111011	$B_4 + B_{18}$	-11111-111
D_{43}	00101011	10001011	$B_{11} + B_{18}$	01111011	$B_4 + B_{18}$	-jjjj1-111
D_{44}	00101100	01000111	$B_7 + B_{17}$	10001011	$B_{11} + B_{18}$	j-j-1-1j-j11
D_{45}	00101101	01110111	$B_4 + B_{17}$	01001011	$B_7 + B_{18}$	-11-j-jj-j11
D_{46}	00101110	10110111	$B_8 + B_{17}$	01111011	$B_4 + B_{18}$	-jj11j-j11
D_{47}	00101111	10000111	$B_{11} + B_{17}$	10111011	$B_8 + B_{18}$	1-1jjj-j11
D_{48}	00110000	11101101	$B_2 + B_{20}$	01000111	$B_7 + B_{17}$	-j1-j-1-j1j1
D_{49}	00110001	10111101	$B_8 + B_{20}$	11100111	$B_2 + B_{17}$	1j1-j-j-j1
D_{50}	00110010	00011101	$B_3 + B_{20}$	10110111	$B_8 + B_{17}$	j-1j1-j1j1
D_{51}	00110011	01001101	$B_7 + B_{20}$	00010111	$B_3 + B_{17}$	-1-j-1j-j1j1
D_{52}	00110100	10101001	$B_{10} + B_{22}$	11001111	$B_3 + B_{16}$	1j-j-11jj1
D_{53}	00110101	00111001	$B_{12} + B_{22}$	10101111	$B_{10} + B_{16}$	j-11-j1jj1

d0 d1 d2 d3 d4 d5 d6 d7		In phase	4-tuple Combination	Quadrature	4-tuple Combination	Complex
D_{54}	00110110	01011001	$B_5 + B_{22}$	00111111	$B_{12} + B_{16}$	-1-jj11jj1
D_{55}	00110111	11001001	$B_3 + B_{22}$	01011111	$B_5 + B_{16}$	-j1-1j1jj1
D_{56}	00111000	00100001	$B_{14} + B_{23}$	10001011	$B_{11} + B_{18}$	j-1-j-1j-1j1
D_{57}	00111001	01110001	$B_4 + B_{23}$	00101011	$B_{14} + B_{18}$	-1-j1-jj-1j1
D_{58}	00111010	11010001	$B_1 + B_{23}$	01111011	$B_4 + B_{18}$	-j1j1j-1j1
D_{59}	00111011	10000001	$B_{11} + B_{23}$	11011011	$B_1 + B_{18}$	1j-1jj-1j1
D_{60}	00111100	01100101	$B_6 + B_{21}$	00000011	$B_{15} + B_{19}$	-1-j-j-1-1-jj1
D_{61}	00111101	11110101	$B_0 + B_{21}$	01100011	$B_6 + B_{19}$	-j11-j-1-jj1
D_{62}	00111110	10010101	$B_9 + B_{21}$	11110011	$B_0 + B_{19}$	1jj1-1-jj1
D_{63}	00111111	00000101	$B_{15} + B_{21}$	10010011	$B_9 + B_{19}$	j-1-1j-1-jj1

IX CCK Decoder

A. Introduction

[0110] A fast correlator transform (FCT) kernel in accordance with the invention, can be used as a building block for a CCK decoder. For example, a FCT kernal can be applied and used to decode data using the IEEE 802.11b 11 Mbps signaling scheme. For the design, two correlators are preferably used. Since the signaling scheme is complex one correlator is used for the in-phase (I) channel, and another correlator is used for the quadrature phase (Q) channel. The input to each channel is the designated codeword for the transmitted data. Table 3 illustrates example input data and corresponding coded output.

Data (input)	Inphase (output)	Quadrature (output)
00000001	00011101 (B13+B20)	11101101 (B2+B20)

Table 3: Input Data and CCK Coded Output

[0111] The CCK coded data for each channel is passed into its respective correlator for processing. The output of a branch correlator trellis preferably has a maximum value of 8 (with no noise in the system) that corresponds to a correlation with the input codeword. The value can be a ∓ 8 depending on the phase. The value is then magnitude squared, which will give one branch of the trellis a maximum value of 64. The in-phase and quadrature branches of the I and Q correlators are then paired together according to the CCK codeword in Table 2, above. These pairs, when combined, provide a maximum I^2+Q^2 value of 128.

[0112] The CCK decoder works fine when there is a 0 or π phase rotation. A problem arises, however, when there is a π/2 or a 3π/2 phase rotation. The problem is described with the following example.

[0113] The base codeword Table 2, above, for the 11Mbps CCK encoding scheme gives the pairs of codewords that correspond to a given input message. For 56 out of the possible 64 codewords, the encoded words on the I and Q channel are similar, except that they are on opposite channels. An example of two codewords are shown in Table 4 below.

Data (input)	Inphase (output)	Quadrature (output)
00001100	11101101 (B2+B20)	00100001 (B14+B23)
00000100	00100001 (B14+B23)	11101101 (B2+B20)

Table 4: CCK Input and Encoded Data

[0114] The codewords for the two data symbols given in Table 4 are similar, but are on different channels. So if there is a π/2 or a 3π/2 phase shift from differentially encoding the transmitted data, then the I and Q channels will be swapped. This is illustrated in FIG. 9, which is a signal path diagram for an example CCK decoder output trellis, including an I signal path 902 and a Q signal path 904.

[0115] If the codewords received on the I signal path 902 and the Q signal path 904 are 11101101 and 00100001, respectively, then the maximum output will be 8 and will follow paths 906a and 906b, respectively. But if the codewords received on the I signal path 902 and the Q signal path 904 are swapped, then the maximum output will be 8 and will follow paths 908a and 908b, respectively, which correspond to different input data than paths 906a and 906b. This is acceptable if that is the intended path. But if there is a π/2 or a 3π/2 phase rotation, then the input words to the correlator are swapped on the I and Q channels, and the I or Q channel is inverted depending on what the phase rotation was. For these cases, if the input codewords before differential encoding correspond to the paths 906a and 906b, then the decoded data should be that which is associated with the paths 906a and 906b, not the paths 908a and 908b. Since the differential decoding process is typically performed after or during the correlation process, then the phase rotated symbols enter the correlator in that form. A determination should be made as to whether there was a phase rotation. Phase rotation can be discerned as follows. If there was a $\pi/2$ phase shift, the in-phase channel will be inverted. If there is a $3\pi/2$ phase shift, the quadrature phase channel will be inverted.

[0116] FIG. 10 is a block diagram of an inverted channel detection circuit 1000 that can be used to determine if one of the channels was inverted. If both channels were inverted then there was a π phase shift and the data is decoded correctly, so this is of no consequence to the system.

X. CCK Correlator Optimizations

A. Introduction

[0117] A CCK parallel correlator in accordance with the invention can be optimized to reduce the number of adders in a Fast Correlator Transform (FCT) kernel. An initial complex correlation is presented below, followed by various optional optimizations that reduce the complexity of the initial complex correlation.

[0118] Table 6 lists 64 complex codewords that have coefficients with four possible values of +1, -1, +j, and –j. This means that either the real part or imaginary part of each coefficient is 0, but not both. Table 6also lists corresponding codewords rotated by 45° to get the I,Q representation. Note that each coefficient in the I,Q representation has a magnitude of $\sqrt{2}$ since the four possible coefficients are now +1+j, +1-j, -1+j, and –1-j. Rotation of the coefficients is the key to simplification of the FCT.

[0119] Additional simplification is achieved by realizing that some intermediate results are the negative of other intermediate results at the same stage.

B. Initial Complex Correlation

[0120] An initial complex correlation of complex CCK codewords with complex inputs can be achieved from equation 12:

$$y(n)=\sum_{k=0}^{7}c^{*}(k)x(n-k) \qquad \text{(Eq. 12)}$$

where $c(k)=c_I(k)+jc_Q(k)$ and $x(k)=x_I(k)+jx_Q(k)$.

[0121] Expanding out the complex product results in equation 13:

$$y(n)=\left[\sum_{k=0}^{7}c_I(k)x_I(n-k)+\sum_{k=0}^{7}c_Q(k)x_Q(n-k)\right]+j\left[\sum_{k=0}^{7}c_I(k)x_Q(n-k)-\sum_{k=0}^{7}c_Q(k)x_I(n-k)\right] \qquad \text{(Eq. 13)}$$

[0122] FIG. 11 is a block diagram 1100 of the complex correlation of equation 12.

[0123] The resulting complex correlation of equation 12 includes four real correlates performed in parallel. Implementing the correlate equations of equation 12 results in 4x7+2=30 additions per codeword. For the 64 CCK codewords there are 64x30=1920 additions to implement all the required correlations. Since the real or imaginary part of each CCK complex coefficient is zero, the computation is reduced to 1920-(64x2x7)=1024.

C. Fast Correlator Transform (FCT)

[0124] A Fast Correlator Transform (FCT) can be used to reduce the computation by assuming the use of 45° rotated I,Q codewords shown in Table 6, below. To derive the 64 parallel correlates for all of the codewords, two substantially similar FCTs are preferred. One is for the real part of the input. The other is for the imaginary part of the input. FIG. 12 is a block diagram of an example FCT 1200. For a CCK with codeword size of 64 the FCT 1200 can be divided into several stages. For example, FIG. 13 illustrates a FCT 1300 that includes stages 1a (1302), 1b (1304), 2a (1306), 2b (1308), and 3 (1310), which are described below.

[0125] FIG. 14 is an example block diagram of stage 1a of the FCT 1300. In the example of FIG. 14, the stage 1a includes 2 additions and 6 subtractions, which implement the following:

$$D_0 = x_0 + x_1,\ D_1 = -x_0 + x_1,\ D_2 = x_0 - x_1,\ D_3 = -x_0 - x_1$$

$$D_4 = x_2 + x_3,\ D_5 = -x_2 + x_3,\ D_6 = x_2 - x_3,\ D_7 = -x_2 - x_3.$$

[0126] FIG. 15 is an example block diagram of stage 1b of the FCT 1300. In the example of FIG. 15, the stage 1b includes 8 additions and subtractions, which implement the following:

$$D_8 = x_4 + x_5,\ D_9 = -x_4 + x_5,\ D_{10} = x_4 - x_5,\ D_{11} = -x_4 - x_5$$

$$D_{12} = x_6 + x_7,\ D_{13} = -x_6 + x_7,\ D_{14} = x_6 - x_7,\ D_{15} = -x_6 - x_7.$$

[0127] FIG. 16 is an example block diagram of stage 2a of the FCT 1300. In the example of FIG. 16, the stage 2a includes 16 additions, which implement the following:

$$B_0 = D_0 + D_4,\ B_1 = D_0 + D_5,\ B_2 = D_0 + D_6,\ B_3 = D_0 + D_7$$

$$B_4 = D_1 + D_4,\ B_5 = D_1 + D_5,\ B_6 = D_1 + D_6,\ B_7 = D_1 + D_7$$

$$B_8 = D_2 + D_4,\ B_9 = D_2 + D_5,\ B_{10} = D_2 + D_6,\ B_{11} = D_2 + D_7$$

$$B_{12} = D_3 + D_4,\ B_{13} = D_3 + D_5,\ B_{14} = D_3 + D_6,\ B_{15} = D_3 + D_7.$$

[0128] FIG. 17 is an example block diagram of stage 2b of the FCT 1300. In the example of FIG. 17, the stage 2b includes 16 additions, which implement the following:

$$B_{16} = D_8 + D_{12},\ B_{17} = D_9 + D_{12},\ B_{18} = D_{10} + D_{12},\ B_{19} = D_{11} + D_{12}$$

$$B_{20} = D_8 + D_{13},\ B_{21} = D_9 + D_{13},\ B_{22} = D_{10} + D_{13},\ B_{23} = D_{11} + D_{13}$$

$$B_{24} = D_8 + D_{14},\ B_{25} = D_9 + D_{14},\ B_{26} = D_{10} + D_{14},\ B_{27} = D_{11} + D_{14}$$

$$B_{28} = D_8 + D_{15},\ B_{29} = D_9 + D_{15},\ B_{30} = D_{10} + D_{15},\ B_{31} = D_{11} + D_{15}$$

[0129] Stage 3 of the FCT 1300 includes 64 additions for the I values and 64 additions for the Q values, as follows:

$$I_0 = B_2 + B_{20},\ I_1 = B_{13} + B_{20},\ I_2 = B_{13} + B_{20},\ I_3 = B_2 + B_{20}$$

$$Q_0 = B_2 + B_{20},\ Q_1 = B_2 + B_{20},\ Q_2 = B_{13} + B_{20},\ Q_3 = B_{13} + B_{20}$$

$$I_4 = B_{14} + B_{23},\ I_5 = B_{13} + B_{23},\ I_6 = B_1 + B_{23},\ I_7 = B_2 + B_{23}$$

$$Q_4 = B_2 + B_{20},\ Q_5 = B_{14} + B_{20},\ Q_6 = B_{13} + B_{20},\ Q_7 = B_1 + B_{20}$$

$$I_8 = B_{14} + B_{23},\ I_9 = B_1 + B_{23},\ I_{10} = B_1 + B_{23},\ I_{11} = B_{14} + B_{23}$$

$$Q_8 = B_{14} + B_{23},\ Q_9 = B_{14} + B_{23},\ Q_{10} = B_1 + B_{23},\ Q_{11} = B_1 + B_{23}$$

$$I_{12} = B_2 + B_{20},\ I_{13} = B_1 + B_{20},\ I_{14} = B_{13} + B_{20},\ I_{15} = B_{14} + B_{20}$$

$$Q_{12} = B_{14} + B_{23}, Q_{13} = B_2 + B_{23}, Q_{14} = B_1 + B_{23},\ Q_{15} = B_{13} + B_{23}$$

$$I_{16} = B_7 + B_{17},\ I_{17} = B_{13} + B_{17},\ I_{18} = B_8 + B_{17},\ I_{19} = B_2 + B_{17}$$

$$Q_{16} = B_2 + B_{20}, Q_{17} = B_7 + B_{20},\ Q_{18} = B_{13} + B_{20},\ Q_{19} = B_8 + B_{20}$$

$$I_{20} = B_{15} + B_{19},\ I_{21} = B_9 + B_{19},\ I_{22} = B_0 + B_{19},\ I_{23} = B_6 + B_{19}$$

$Q_{20} = B_6 + B_{21}$, $Q_{21} = B_{15} + B_{21}$, $Q_{22} = B_9 + B_{21}$, $Q_{23} = B_0 + B_{21}$

$I_{24} = B_{11} + B_{18}$, $I_{25} = B_1 + B_{18}$, $I_{26} = B_4 + B_{18}$, $I_{27} = B_{14} + B_{18}$

$Q_{24} = B_{14} + B_{23}$, $Q_{25} = B_{11} + B_{23}$, $Q_{26} = B_1 + B_{23}$, $Q_{27} = B_4 + B_{23}$

$I_{28} = B_3 + B_{16}$, $I_{29} = B_5 + B_{16}$, $I_{30} = B_{12} + B_{16}$, $I_{31} = B_{10} + B_{16}$

$Q_{28} = B_{10} + B_{22}$, $Q_{29} = B_3 + B_{22}$, $Q_{30} = B_5 + B_{22}$, $Q_{31} = B_{12} + B_{22}$

$I_{32} = B_7 + B_{17}$, $I_{33} = B_8 + B_{17}$, $I_{34} = B_8 + B_{17}$, $I_{35} = B_7 + B_{17}$

$Q_{32} = B_7 + B_{17}$, $Q_{33} = B_7 + B_{17}$, $Q_{34} = B_8 + B_{17}$, $Q_{35} = B_8 + B_{17}$

$I_{36} = B_{11} + B_{18}$, $I_{37} = B_8 + B_{18}$, $I_{38} = B_4 + B_{18}$, $I_{39} = B_7 + B_{18}$

$Q_{36} = B_7 + B_{17}$, $Q_{37} = B_{11} + B_{17}$, $Q_{38} = B_8 + B_{17}$, $Q_{39} = B_4 + B_{17}$

$I_{40} = B_{11} + B_{18}$, $I_{41} = B_4 + B_{18}$, $I_{42} = B_4 + B_{18}$, $I_{43} = B_{11} + B_{18}$

$Q_{40} = B_{11} + B_{18}$, $Q_{41} = B_{11} + B_{18}$, $Q_{42} = B_4 + B_{18}$, $Q_{43} = B_4 + B_{18}$

$I_{44} = B_7 + B_{17}$, $I_{45} = B_4 + B_{17}$, $I_{46} = B_{11} + B_{17}$, $I_{47} = B_{11} + B_{17}$

$Q_{44} = B_{11} + B_{18}$, $Q_{45} = B_7 + B_{18}$, $Q_{46} = B_4 + B_{18}$, $Q_{47} = B_8 + B_{18}$

$I_{48} = B_2 + B_{20}$, $I_{49} = B_8 + B_{20}$, $I_{50} = B_{13} + B_{20}$, $I_{51} = B_7 + B_{20}$

$Q_{48} = B_7 + B_{17}$, $Q_{49} = B_2 + B_{17}$, $Q_{50} = B_8 + B_{17}$, $Q_{51} = B_{13} + B_{17}$

$I_{52} = B_{10} + B_{22}$, $I_{53} = B_{12} + B_{22}$, $I_{54} = B_5 + B_{22}$, $I_{55} = B_3 + B_{22}$

$Q_{52} = B_3 + B_{16}$, $Q_{53} = B_{10} + B_{16}$, $Q_{54} = B_{12} + B_{16}$, $Q_{55} = B_5 + B_{16}$

$I_{56} = B_{14} + B_{23}$, $I_{57} = B_4 + B_{23}$, $I_{58} = B_1 + B_{22}$, $I_{59} = B_{11} + B_{23}$

$Q_{56} = B_{11} + B_{18}$, $Q_{57} = B_{14} + B_{18}$, $Q_{58} = B_4 + B_{18}$, $Q_{59} = B_1 + B_{18}$

$I_{60} = B_6 + B_{21}$, $I_{61} = B_0 + B_{21}$, $I_{62} = B_9 + B_{21}$, $I_{63} = B_{15} + B_{21}$

$$Q_{60} = B_{15} + B_{19}, Q_{61} = B_6 + B_{19}, Q_{62} = B_0 + B_{19}, Q_{63} = B_9 + B_{19}$$

[0130] Stage 3 can be optimized by noting that there are 40 distinct values for I and Q which are

$$A_0 = B_2 + B_{20}, A_1 = B_{13} + B_{20}, A_2 = B_{14} + B_{23}, A_3 = B_{13} + B_{23}$$

$$A_4 = B_{14} + B_{20}, A_5 = B_1 + B_{23}, A_6 = B_2 + B_{23}, A_7 = B_1 + B_{20}$$

$$A_8 = B_7 + B_{17}, A_9 = B_{13} + B_{17}, A_{10} = B_7 + B_{20}, A_{11} = B_8 + B_{17}$$

$$A_{12} = B_2 + B_{17}, A_{13} = B_8 + B_{20}, A_{14} = B_{15} + B_{19}, A_{15} = B_6 + B_{21}$$

$$A_{16} = B_9 + B_{19}, A_{17} = B_{15} + B_{21}, A_{18} = B_0 + B_{19}, A_{19} = B_9 + B_{21}$$

$$A_{20} = B_6 + B_{19}, A_{21} = B_0 + B_{21}, A_{22} = B_{11} + B_{18}, A_{23} = B_1 + B_{18}$$

$$A_{24} = B_{11} + B_{23}, A_{25} = B_4 + B_{18}, A_{26} = B_{14} + B_{18}, A_{27} = B_4 + B_{23}$$

$$A_{28} = B_3 + B_{16}, A_{29} = B_{10} + B_{22}\ A_{30} = B_5 + B_{16}, A_{31} = B_3 + B_{22}$$

$$A_{32} = B_{12} + B_{16}, A_{33} = B_5 + B_{22}, A_{34} = B_{10} + B_{16}, A_{35} = B_{12} + B_{22},$$

$$A_{36} = B_8 + B_{18}, A_{37} = B_{11} + B_{17}, A_{38} = B_7 + B_{18}, A_{39} = B_4 + B_{17}.$$

[0131] FIG. 18 is a block diagram of an example system 1800 that calculates the 40 A_n values. The 40 A_n values are used to calculate the final 64 output values and are described below.

[0132] Further optimization of the FCT 1300 can be achieved by considering the relationship:

$$B_n = -B_{15-n} \quad ,8 \le n \le 15 \qquad \text{(Eq. 14)}$$

[0133] The relationship of equation 14 eliminates the need to calculate the eight values of B_n for $8 \le n \le 15$, and the intermediate results D_2 and D_3. Thus, 8+2=10 additions can be removed and be replaced by 10 negates of the samples, or subtraction instead of addition at the next processing stage.

[0134] Because c_7 is always 1, calculation of $B_{24} \cdots B_{31}$ can eliminate the following eight additions:

$$B_{24} = D_8 + D_{14},\ B_{25} = D_9 + D_{14},\ B_{26} = D_{10} + D_{14},\ B_{27} = D_{11} + D_{14}$$

$$B_{28} = D_8 + D_{15},\ B_{29} = D_9 + D_{15},\ B_{30} = D_{10} + D_{15},\ B_{31} = D_{11} + D_{15}$$

and, consequently, the following two additions:

$$D_{14} = x_6 - x_7,\ D_{15} = -x_6 - x_7.$$

[0135] Four more additions can be removed by considering that:

$$D_6 = -D_5,\ D_7 = -D_4,\ D_{10} = -D_9,\ D_{11} = -D_8$$

[0136] Table 6 shows the 64 correlate outputs as a function of $B_0 \cdots B_7$ and $B_{16} \cdots B_{23}$. Note that 40 distinct summations, or subtractions, are used to calculate the 128 results in the final stage. FIG. 19 is a block diagram of connection stages 1a, 1b, 2a, 2b, and 3 for an example simplified architecture FCT 1900. Note the reduction in states between stages from the FCT 1300 in FIG. 13 and the FCT 1900 in FIG. 19. The FCT 1900 includes stages 1a (1902), 1b (1904), 2a (1906), 2c (1908), and 3 (1910), which are described below.

[0137] FIG. 20 is an example block diagram of the stage 1a of the FCT 1900. In the example of FIG. 20, stage 1a includes 4 additions and subtractions, which implement the following:

$$D_0 = x_0 + x_1,\ D_1 = -x_0 + x_1$$

$$D_4 = x_2 + x_3,\ D_5 = -x_2 + x_3$$

[0138] FIG. 21 is an example block diagram of the stage 1b of the FCT 1900. In the example of FIG. 21, stage 1b includes 4 additions or subtractions, which implement the following:

$$D_8 = x_4 + x_5,\ D_9 = -x_4 + x_5$$

$D_{12} = x_6 + x_7$, $D_{13} = -x_6 + x_7$

[0139] FIG. 22 is an example block diagram of stage 2a of the FCT 1900. In the example of FIG. 22, stage 2a includes 8 additions or subtractions, which implement the following:

$$B_0 = D_0 + D_4,\ B_1 = D_0 + D_5,\ B_2 = D_0 - D_5,\ B_3 = D_0 - D_4$$

$$B_4 = D_1 + D_4,\ B_5 = D_1 + D_5,\ B_6 = D_1 - D_5,\ B_7 = D_1 - D_4$$

[0140] FIG. 23 is an example block diagram of stage 2b of the FCT 1900. In the example of FIG. 23, stage 2b includes 8 additions, which implement the following:

$$B_{16} = D_8 + D_{12},\quad B_{17} = D_9 + D_{12},\quad B_{18} = -D_9 + D_{12},\quad B_{19} = -D_8 + D_{12}$$

$$B_{20} = D_8 + D_{13},\quad B_{21} = D_9 + D_{13},\quad B_{22} = -D_9 + D_{13},\quad B_{23} = -D_8 + D_{13}$$

[0141] FIGS. 24A and 24B illustrate an example block diagram of stage 3 of the FCT 1900. In the example of FIGS. 24A and 24B, stage 3 includes 40 additions or subtractions, which implement the following:

$$A_0 = B_2 + B_{20},\ A_1 = -B_2 + B_{20},\ A_2 = -B_1 + B_{23},\ A_3 = -B_2 + B_{23}$$

$$A_4 = -B_1 + B_{20},\ A_5 = B_1 + B_{23},\ A_6 = B_2 + B_{23},\ A_7 = B_1 + B_{20}$$

$$A_8 = B_7 + B_{17},\ A_9 = -B_2 + B_{17},\ A_{10} = B_7 + B_{20},\ A_{11} = -B_7 + B_{17}$$

$$A_{12} = B_2 + B_{17},\ A_{13} = -B_7 + B_{20},\ A_{14} = -B_0 + B_{19},\ A_{15} = B_6 + B_{21}$$

$$A_{16} = -B_6 + B_{19},\ A_{17} = -B_0 + B_{21},\ A_{18} = B_0 + B_{19},\ A_{19} = -B_6 + B_{21}$$

$$A_{20} = B_6 + B_{19},\ A_{21} = B_0 + B_{21},\ A_{22} = -B_4 + B_{18},\ A_{23} = B_1 + B_{18}$$

$$A_{24} = -B_4 + B_{23},\ A_{25} = B_4 + B_{18},\ A_{26} = -B_1 + B_{18},\ A_{27} = B_4 + B_{23}$$

$A_{28} = B_3 + B_{16}$, $A_{29} = -B_5 + B_{22}$ $A_{30} = B_5 + B_{16}$, $A_{31} = B_3 + B_{22}$

$A_{32} = -B_3 + B_{16}$, $A_{33} = B_5 + B_{22}$, $A_{34} = -B_5 + B_{16}$, $A_{35} = -B_3 + B_{22}$

$A_{36} = -B_7 + B_{18}$, $A_{37} = -B_4 + B_{17}$, $A_{38} = B_7 + B_{18}$, $A_{39} = B_4 + B_{17}$

[0142] Final outputs are calculated using 128 additions or subtractions.

$Y_{I0} = A_{I0} + A_{Q0}$, $Y_{Q0} = A_{Q0} - A_{I0}$, $Y_{I1} = A_{I1} + A_{Q0}$, $Y_{Q1} = A_{Q1} - A_{I0}$

$Y_{I2} = A_{I1} + A_{Q1}$, $Y_{Q2} = A_{Q1} - A_{I1}$, $Y_{I3} = A_{I0} + A_{Q1}$, $Y_{Q3} = A_{Q0} - A_{I1}$

$Y_{I4} = A_{I2} + A_{Q0}$, $Y_{Q4} = A_{Q2} - A_{I0}$, $Y_{I5} = A_{I3} + A_{Q4}$, $Y_{Q5} = A_{Q3} - A_{I4}$

$Y_{I6} = A_{I5} + A_{Q1}$, $Y_{Q6} = A_{Q5} - A_{I1}$, $Y_{I7} = A_{I6} + A_{Q7}$, $Y_{Q7} = A_{Q6} - A_{I7}$

$Y_{I8} = A_{I2} + A_{Q2}$, $Y_{Q8} = A_{Q2} - A_{I2}$, $Y_{I9} = A_{I5} + A_{Q2}$, $Y_{Q9} = A_{Q5} - A_{I2}$

$Y_{I10} = A_{I5} + A_{Q5}$, $Y_{Q10} = A_{Q5} - A_{I5}$, $Y_{I11} = A_{I2} + A_{Q5}$, $Y_{Q11} = A_{Q2} - A_{I5}$

$Y_{I12} = A_{I0} + A_{Q2}$, $Y_{Q12} = A_{Q0} - A_{I2}$, $Y_{I13} = A_{I7} + A_{Q6}$, $Y_{Q13} = A_{Q7} - A_{I6}$

$Y_{I14} = A_{I1} + A_{Q5}$, $Y_{Q14} = A_{Q1} - A_{I5}$, $Y_{I15} = A_{I4} + A_{Q3}$, $Y_{Q15} = A_{Q4} - A_{I3}$

$Y_{I16} = A_{I8} + A_{Q0}$, $Y_{Q16} = A_{Q8} - A_{I0}$, $Y_{I17} = A_{I9} + A_{Q10}$, $Y_{Q17} = A_{Q9} - A_{I10}$

$Y_{I18} = A_{I11} + A_{Q1}$, $Y_{Q18} = A_{Q11} - A_{I1}$, $Y_{I19} = A_{I12} + A_{Q13}$, $Y_{Q19} = A_{Q12} - A_{I13}$

$Y_{I20} = A_{I14} + A_{Q15}$, $Y_{Q20} = A_{Q14} - A_{I15}$, $Y_{I21} = A_{I16} + A_{Q17}$,
$Y_{Q21} = A_{Q16} - A_{I17}$

$Y_{I22} = A_{I18} + A_{Q19}$, $Y_{Q22} = A_{Q18} - A_{I19}$, $Y_{I23} = A_{I20} + A_{Q21}$,
$Y_{Q23} = A_{Q20} - A_{I21}$

$Y_{I24} = A_{I22} + A_{Q2}$, $Y_{Q24} = A_{Q22} - A_{I2}$, $Y_{I25} = A_{I23} + A_{Q24}$,
$Y_{Q25} = A_{Q23} - A_{I24}$

$Y_{I26} = A_{I25} + A_{Q5}$, $Y_{Q26} = A_{Q25} - A_{I5}$, $Y_{I27} = A_{I26} + A_{Q27}$,
$Y_{Q27} = A_{Q26} - A_{I27}$

$Y_{I28} = A_{I28} + A_{Q29}$, $Y_{Q28} = A_{Q28} - A_{I29}$, $Y_{I29} = A_{I30} + A_{Q31}$,
$Y_{Q29} = A_{Q30} - A_{I31}$

$Y_{I30} = A_{I32} + A_{Q33}$, $Y_{Q30} = A_{Q32} - A_{I33}$, $Y_{I31} = A_{I34} + A_{Q35}$,
$Y_{Q31} = A_{Q34} - A_{I35}$

$Y_{I32} = A_{I8} + A_{Q8}$, $Y_{Q32} = A_{Q8} - A_{I8}$, $Y_{I33} = A_{I11} + A_{Q8}$,
$Y_{Q33} = A_{Q11} - A_{I8}$

$Y_{I34} = A_{I11} + A_{Q11}$, $Y_{Q34} = A_{Q11} - A_{I11}$, $Y_{I35} = A_{I8} + A_{Q11}$,
$Y_{Q35} = A_{Q8} - A_{I11}$

$Y_{I36} = A_{I22} + A_{Q8}$, $Y_{Q36} = A_{Q22} - A_{I8}$, $Y_{I37} = A_{I36} + A_{Q37}$,
$Y_{Q37} = A_{Q36} - A_{I37}$

$Y_{I38} = A_{I25} + A_{Q11}$, $Y_{Q38} = A_{Q25} - A_{I11}$, $Y_{I39} = A_{I38} + A_{Q39}$,
$Y_{Q39} = A_{Q38} - A_{I39}$

$Y_{I40} = A_{I22} + A_{Q22}$, $Y_{Q40} = A_{Q22} - A_{I22}$, $Y_{I41} = A_{I25} + A_{Q22}$,
$Y_{Q41} = A_{Q25} - A_{I22}$

$Y_{I42} = A_{I25} + A_{Q25}$, $Y_{Q42} = A_{Q25} - A_{I25}$, $Y_{I43} = A_{I22} + A_{Q25}$,
$Y_{Q43} = A_{Q22} - A_{I25}$

$Y_{I44} = A_{I8} + A_{Q22}$, $Y_{Q44} = A_{Q8} - A_{I22}$, $Y_{I45} = A_{I39} + A_{Q38}$,
$Y_{Q45} = A_{Q39} - A_{I38}$

$Y_{I46} = A_{I11} + A_{Q25}$, $Y_{Q46} = A_{Q11} - A_{I25}$, $Y_{I47} = A_{I37} + A_{Q36}$,
$Y_{Q47} = A_{Q37} - A_{I36}$

$Y_{I48} = A_{I0} + A_{Q8}$, $Y_{Q48} = A_{Q0} - A_{I8}$, $Y_{I49} = A_{I13} + A_{Q12}$,
$Y_{Q49} = A_{Q13} - A_{I12}$

$Y_{I50} = A_{I1} + A_{Q11}$, $Y_{Q50} = A_{Q1} - A_{I11}$, $Y_{I51} = A_{I10} + A_{Q9}$,
$Y_{Q51} = A_{Q10} - A_{I9}$

$Y_{I52} = A_{I29} + A_{Q28}$, $Y_{Q52} = A_{Q29} - A_{28}$, $Y_{I53} = A_{I35} + A_{Q34}$,
$Y_{Q53} = A_{Q35} - A_{34}$

$Y_{I54} = A_{I33} + A_{Q32}$, $Y_{Q54} = A_{Q33} - A_{32}$, $Y_{I55} = A_{I31} + A_{Q30}$,
$Y_{Q55} = A_{Q31} - A_{30}$

$Y_{I56} = A_{I2} + A_{Q22}$, $Y_{Q56} = A_{Q2} - A_{22}$, $Y_{I57} = A_{I27} + A_{Q26}$,
$Y_{Q57} = A_{Q27} - A_{26}$

$Y_{I58} = A_{I5} + A_{Q25}$, $Y_{Q58} = A_{Q5} - A_{25}$, $Y_{I59} = A_{I24} + A_{Q23}$,
$Y_{Q59} = A_{Q24} - A_{23}$

$Y_{I60} = A_{I15} + A_{Q14}$, $Y_{Q60} = A_{Q15} - A_{14}$, $Y_{I61} = A_{I21} + A_{Q20}$,
$Y_{Q61} = A_{Q21} - A_{20}$

$Y_{I62} = A_{I19} + A_{Q18}$, $Y_{Q62} = A_{Q19} - A_{18}$, $Y_{I63} = A_{I17} + A_{Q16}$,
$Y_{Q63} = A_{Q17} - A_{16}$

[0143] The number of adders in the final stage can be reduced by almost half from 128 to 72 by sharing adders. The adders are shared between codewords that are complex conjugates pairs as listed in Table 5. There are a total of 36 complex conjugate pairs where 8 of the codewords are complex conjugates of themselves (real coefficients). Consider, as an example, the pair of codeword 1 and codeword 3, from Table 5, with the following output equations:

$$Y_{I1} = A_{I1} + A_{Q0},\ Y_{Q1} = A_{Q1} - A_{I0} \qquad \text{(Eq. 15)}$$

$$Y_{I3} = A_{I0} + A_{Q1},\ Y_{Q3} = A_{Q0} - A_{I1} \qquad \text{(Eq. 16)}$$

[0144] The four additions, or subtractions, can be reduced to two additions by using the architecture shown in FIG. 18. A control signal sets the branches shown to opposite signs and the registers are loaded on opposite edges of the clock. Outputs include four output values stored in the four registers. This structure is repeated 36 times for each of the conjugate a, b codeword pairs in Table 5.

[0145] To summarize, the total number of complex additions, or subtractions, for the simplified parallel correlator 1900 are:

Stage #1 = 8

Stage #2 = 16

Stage #3 = 40+36 = 76

TOTAL = 100

Codeword Index a	**Codeword Index b**	**Codeword Index a**	**Codeword Index b**	**Codeword Index a**	**Codewor d Index b**
0	---	16	48	32	---
1	3	17	51	33	35
2	---	18	50	34	---
4	12	19	49	36	44
5	15	20	60	37	47
6	14	21	63	38	46
7	13	22	62	39	45
8	---	23	61	40	---
9	11	24	56	41	43
10	---	25	59	42	---
		26	58		
		27	57		
		28	52		
		29	55		
		30	54		
		31	53		

Table 5, Codeword Conjugate Pairs

D. FCT Based on CCK Code Properties

[0146] Another less complex embodiment can be derived by considering the equations for generating the CCK correlation output:

$$d = c_7 x_7 + c_6 x_6 + c_5 x_5 + c_4 x_4 + c_3 x_3 + c_2 x_2 + c_1 x_1 + c_0 x_0 \qquad \text{(Eq. 16)}$$

[0147] where $c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0$ are the complex coefficients and $x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0$ are the complex input buffer samples. The codewords can be uniquely defined by the three coefficients, c_6, c_5, c_3, with the other coefficients defined by

$$c_0 = c_6 c_5 c_3$$

$$c_1 = c_5 c_3$$

$$c_2 = c_6 c_3$$

$$c_4 = -c_6 c_5$$

$$c_7 = 1$$

[0148] Thus, after substituting coefficient relationships and rearranging, the correlation output becomes:

$$d = x_7 + c_6 x_6 + c_5 (x_5 - c_6 x_4) + c_3 \{x_3 + c_6 x_2 + c_5 [x_1 + c_6 x_0]\} \text{(Eq. 17)}$$

[0149] For correlation with a single codeword, the resulting structure for equation 170 is shown in FIG. 25.

[0150] The FCT is derived from the single correlate by converting each coefficient branch, starting at the input, to a positive and negative branch. The resulting structure is shown in FIGS. 26 and 27.

[0151] Stage 1 consists of 4 additions or subtractions.

$$D_0 = x_0 + x_1,\ D_1 = -x_0 + x_1,\ D_2 = x_2,\ D_3 = x_3$$

$$D_4 = x_4 + x_5,\ D_5 = -x_4 + x_5,\ D_6 = x_6,\ D_7 = x_7$$

[0152] Stage 2 consists of 8 additions or subtractions.

$$E_0 = D_0 + D_2,\ E_1 = D_2 - D_0,\ E_2 = D_1 + D_2,\ E_3 = D_1 - D_2$$

$$E_4 = D_4 + D_6,\ E_5 = D_6 - D_4,\ E_6 = D_5 + D_6,\ E_7 = D_5 - D_6$$

[0153] Stage 3 consists of 16 additions or subtractions.

$$B_0 = E_0 + D_3,\ -B_3 = E_1 + D_3,\ -B_2 = -E_0 + D_3,\ B_1 = -E_1 + D_3$$

$$B_4 = E_2 + D_3,\ B_5 = E_3 + D_3,\ -B_6 = -E_2 + D_3,\ -B_7 = -E_3 + D_3$$

$$B_{16} = E_4 + D_7,\ B_{18} = E_5 + D_7,\ B_{23} = -E_4 + D_7,\ B_{20} = -E_5 + D_7$$

$$B_{17} = E_6 + D_7,\ B_{21} = E_7 + D_7,\ B_{22} = -E_6 + D_7,\ B_{18} = -E_7 + D_7$$

[0154] Stage 4 consists of 40 additions or subtractions.

$$A_0 = B_2 + B_{20},\ A_1 = -B_2 + B_{20},\ A_2 = -B_1 + B_{23},\ A_3 = -B_2 + B_{23}$$

$$A_4 = -B_1 + B_{20},\ A_5 = B_1 + B_{23},\ A_6 = B_2 + B_{23},\ A_7 = B_1 + B_{20}$$

$$A_8 = B_7 + B_{17},\ A_9 = -B_2 + B_{17},\ A_{10} = B_7 + B_{20},\ A_{11} = -B_7 + B_{17}$$

$$A_{12} = B_2 + B_{17},\ A_{13} = -B_7 + B_{20},\ A_{14} = -B_0 + B_{19},\ A_{15} = B_6 + B_{21}$$

$$A_{16} = -B_6 + B_{19},\ A_{17} = -B_0 + B_{21},\ A_{18} = B_0 + B_{19},\ A_{19} = -B_6 + B_{21}$$

$$A_{20} = B_6 + B_{19},\ A_{21} = B_0 + B_{21},\ A_{22} = -B_4 + B_{18},\ A_{23} = B_1 + B_{18}$$

$$A_{24} = -B_4 + B_{23},\ A_{25} = B_4 + B_{18},\ A_{26} = -B_1 + B_{18},\ A_{27} = B_4 + B_{23}$$

$$A_{28} = B_3 + B_{16},\ A_{29} = -B_5 + B_{22}\ A_{30} = B_5 + B_{16},\ A_{31} = B_3 + B_{22}$$

$$A_{32} = -B_3 + B_{16},\qquad A_{33} = B_5 + B_{22},\qquad A_{34} = -B_5 + B_{16},\ A_{35} = -B_3 + B_{22}$$

$$A_{36} = -B_7 + B_{18},\ A_{37} = -B_4 + B_{17},\ A_{38} = B_7 + B_{18},\ A_{39} = B_4 + B_{17}$$

Table 6

Index	Bits	Complex Codeword	I,Q Codeword	I,Q Combo
0	000000	+1+1+1-1+1+1-1+1	+1+1+1-1+1+1-1+1 +1+1+1-1+1+1-1+1	B2+B20 B2+B20
1	000001	+j+j+j-j+1+1-1+1	-1-1-1+1+1+1-1+1 +1+1+1-1+1+1-1+1	-B2+B20 B2+B20
2	000010	-1-1-1+1+1+1-1+1	-1-1-1+1+1+1-1+1 -1-1-1+1+1+1-1+1	-B2+B20 -B2+B20
3	000011	-j-j-j+j+1+1-1+1	+1+1+1-1+1+1-1+1 -1-1-1+1+1+1-1+1	B2+B20 -B2+B20
4	000100	+j+j+1-1+j+j-1+1	-1-1+1-1-1-1-1+1 +1+1+1-1+1+1-1+1	-B1+B23 B2+B20
5	000101	-1-1+j-j+j+j-1+1	-1-1-1+1-1-1-1+1 -1-1+1-1+1+1-1+1	-B2+B23 -B1+B20
6	000110	-j-j-1+1+j+j-1+1	+1+1-1+1-1-1-1+1 -1-1-1+1+1+1-1+1	B1+B23 -B2+B20
7	000111	+1+1-j+j+j+j-1+1	+1+1+1-1-1-1-1+1 +1+1-1+1+1+1-1+1	B2+B23 B1+B20
8	001000	-1-1+1-1-1-1-1+1	-1-1+1-1-1-1-1+1 -1-1+1-1-1-1-1+1	-B1+B23 -B1+B23
9	001001	-j-j+j-j-1-1-1+1	+1+1-1+1-1-1-1+1 -1-1+1-1-1-1-1+1	B1+B23 -B1+B23
10	001010	+1+1-1+1-1-1-1+1	+1+1-1+1-1-1-1+1 +1+1-1+1-1-1-1+1	B1+B23 B1+B23
11	001011	+j+j-j+j-1-1-1+1	-1-1+1-1-1-1-1+1 +1+1-1+1-1-1-1+1	-B1+B23 B1+B23
12	001100	-j-j+1-1-j-j-1+1	+1+1+1-1+1+1-1+1 -1-1+1-1-1-1-1+1	B2+B20 -B1+B23
13	001101	+1+1+j-j-j-j-1+1	+1+1-1+1+1+1-1+1 +1+1+1-1-1-1-1+1	B1+B20 B2+B23
14	001110	+j+j-1+1-j-j-1+1	-1-1-1+1+1+1-1+1 +1+1-1+1-1-1-1+1	-B2+B20 B1+B23
15	001111	-1-1-j+j-j-j-1+1	-1-1+1-1+1+1-1+1 -1-1-1+1-1-1-1+1	-B1+B20 -B2+B23

Table 6 (continued)

Index	Bits	Complex Codeword	I,Q Codeword	I,Q Combo
16	010000	+j+1+j-1+j+1-j+1	-1+1-1-1-1+1+1+1 +1+1+1-1+1+1-1+1	B7+B17 B2+B20
17	010001	-1+j-1-j+j+1-j+1	-1-1-1+1-1+1+1+1 -1+1-1-1+1+1-1+1	-B2+B17 B7+B20
18	010010	-j-1-j+1+j+1-j+1	+1-1+1+1-1+1+1+1 -1-1-1+1+1+1-1+1	-B7+B17 -B2+B20
19	010011	+1-j+1+j+j+1-j+1	+1+1+1-1-1+1+1+1 +1-1+1+1+1+1-1+1	B2+B17 -B7+B20
20	010100	-1+j+j-1-1+j-j+1	-1-1-1-1-1-1+1+1 -1+1+1-1-1+1-1+1	-B0+B19 B6+B21
21	010101	-j-1-1-j-1+j-j+1	+1-1-1+1-1-1+1+1 -1-1-1-1-1+1-1+1	-B6+B19 -B0+B21
22	010110	+1-j-j+1-1+j-j+1	+1+1+1+1-1-1+1+1 +1-1-1+1-1+1-1+1	B0+B19 -B6+B21
23	010111	+j+1+1+j-1+j-j+1	-1+1+1-1-1-1+1+1 +1+1+1+1-1+1-1+1	B6+B19 B0+B21
24	011000	-j-1+j-1-j-1-j+1	+1-1-1-1+1-1+1+1 -1-1+1-1-1-1-1+1	-B4+B18 -B1+B23
25	011001	+1-j-1-j-j-1-j+1	+1+1-1+1+1-1+1+1 +1-1-1-1-1-1-1+1	B1+B18 -B4+B23
26	011010	+j+1-j+1-j-1-j+1	-1+1+1+1+1-1+1+1 +1+1-1+1-1-1-1+1	B4+B18 B1+B23
27	011011	-1+j+1+j-j-1-j+1	-1-1+1-1+1-1+1+1 -1+1+1+1-1-1-1+1	-B1+B18 B4+B23
28	011100	+1-j+j-1+1-j-j+1	+1+1-1-1+1+1+1+1 +1-1+1-1+1-1-1+1	B3+B16 -B5+B22
29	011101	+j+1-1-j+1-j-j+1	-1+1-1+1+1+1+1+1 +1+1-1-1+1-1-1+1	B5+B16 B3+B22
30	011110	-1+j-j+1+1-j-j+1	-1-1+1+1+1+1+1+1 -1+1-1+1+1-1-1+1	-B3+B16 B5+B22
31	011111	-j-1+1+j+1-j-j+1	+1-1+1-1+1+1+1+1 -1-1+1+1+1-1-1+1	-B5+B16 -B3+B22

Table 6 (continued)

Index	Bits	Complex Codeword	I,Q Codeword	I,Q Combo
32	100000	-1+1-1-1-1+1+1+1	-1+1-1-1-1+1+1+1 -1+1-1-1-1+1+1+1	B7+B17 B7+B17
33	100001	-j+j-j-j-1+1+1+1	+1-1+1+1-1+1+1+1 -1+1-1-1-1+1+1+1	-B7+B17 B7+B17
34	100010	+1-1+1+1-1+1+1+1	+1-1+1+1-1+1+1+1 +1-1+1+1-1+1+1+1	-B7+B17 -B7+B17
35	100011	+j-j+j+j-1+1+1+1	-1+1-1-1-1+1+1+1 +1-1+1+1-1+1+1+1	B7+B17 -B7+B17
36	100100	-j+j-1-1-j+j+1+1	+1-1-1-1+1-1+1+1 -1+1-1-1-1+1+1+1	-B4+B18 B7+B17
37	100101	+1-1-j-j-j+j+1+1	+1-1+1+1+1-1+1+1 +1-1-1-1-1+1+1+1	-B7+B18 -B4+B17
38	100110	+j-j+1+1-j+j+1+1	-1+1+1+1+1-1+1+1 +1-1+1+1-1+1+1+1	B4+B18 -B7+B17
39	100111	-1+1+j+j-j+j+1+1	-1+1-1-1+1-1+1+1 -1+1+1+1-1+1+1+1	B7+B18 B4+B17
40	101000	+1-1-1-1+1-1+1+1	+1-1-1-1+1-1+1+1 +1-1-1-1+1-1+1+1	-B4+B18 -B4+B18
41	101001	+j-j-j-j+1-1+1+1	-1+1+1+1+1-1+1+1 +1-1-1-1+1-1+1+1	B4+B18 -B4+B18
42	101010	-1+1+1+1+1-1+1+1	-1+1+1+1+1-1+1+1 -1+1+1+1+1-1+1+1	B4+B18 B4+B18
43	101011	-j+j+j+j+1-1+1+1	+1-1-1-1+1-1+1+1 -1+1+1+1+1-1+1+1	-B4+B18 B4+B18
44	101100	+j-j-1-1+j-j+1+1	-1+1-1-1-1+1+1+1 +1-1-1-1+1-1+1+1	B7+B17 -B4+B18
45	101101	-1+1-j-j+j-j+1+1	-1+1+1+1-1+1+1+1 -1+1-1-1+1-1+1+1	B4+B17 B7+B18
46	101110	-j+j+1+1+j-j+1+1	+1-1+1+1-1+1+1+1 -1+1+1+1+1-1+1+1	-B7+B17 B4+B18
47	101111	+1-1+j+j+j-j+1+1	+1-1-1-1-1+1+1+1 +1-1+1+1+1-1+1+1	-B4+B17 -B7+B18

Table 6 (continued)

Index	Bits	Complex Codeword	I,Q Codeword	I,Q Combo
48	110000	-j+1-j-1-j+1+j+1	+1+1+1-1+1+1-1+1 -1+1-1-1-1+1+1+1	B2+B20 B7+B17
49	110001	+1+j+1-j-j+1+j+1	+1-1+1+1+1+1-1+1 +1+1+1-1-1+1+1+1	-B7+B20 B2+B17
50	110010	+j-1+j+1-j+1+j+1	-1-1-1+1+1+1-1+1 +1-1+1+1-1+1+1+1	-B2+B20 -B7+B17
51	110011	-1-j-1+j-j+1+j+1	-1+1-1-1+1+1-1+1 -1-1-1+1-1+1+1+1	B7+B20 -B2+B17
52	110100	+1+j-j-1+1+j+j+1	+1-1+1-1+1-1-1+1 +1+1-1-1+1+1+1+1	-B5+B22 B3+B16
53	110101	+j-1+1-j+1+j+j+1	-1-1+1+1+1-1-1+1 +1-1+1-1+1+1+1+1	-B3+B22 -B5+B16
54	110110	-1-j+j+1+1+j+j+1	-1+1-1+1+1-1-1+1 -1-1+1+1+1+1+1+1	B5+B22 -B3+B16
55	110111	-j+1-1+j+1+j+j+1	+1+1-1-1+1-1-1+1 -1+1-1+1+1+1+1+1	B3+B22 B5+B16
56	111000	+j-1-j-1+j-1+j+1	-1-1+1-1-1-1-1+1 +1-1-1-1+1-1+1+1	-B1+B23 -B4+B18
57	111001	-1-j+1-j+j-1+j+1	-1+1+1+1-1-1-1+1 -1-1+1-1+1-1+1+1	B4+B23 -B1+B18
58	111010	-j+1+j+1+j-1+j+1	+1+1-1+1-1-1-1+1 -1+1+1+1+1-1+1+1	B1+B23 B4+B18
59	111011	+1+j-1+j+j-1+j+1	+1-1-1-1-1-1-1+1 +1+1-1+1+1-1+1+1	-B4+B23 B1+B18
60	111100	-1-j-j-1-1-j+j+1	-1+1+1-1-1+1-1+1 -1-1-1-1-1-1+1+1	B6+B21 -B0+B19
61	111101	-j+1+1-j-1-j+j+1	+1+1+1+1-1+1-1+1 -1+1+1-1-1-1+1+1	B0+B21 B6+B19
62	111110	+1+j+j+1-1-j+j+1	+1-1-1+1-1+1-1+1 +1+1+1+1-1-1+1+1	-B6+B21 B0+B19
63	111111	+j-1-1+j-1-j+j+1	-1-1-1-1-1+1-1+1 +1-1-1+1-1-1+1+1	-B0+B21 -B6+B19

XI. Conclusion

[0155] While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. Other embodiments are possible and are covered by the invention.